

Reachability and Termination Analysis of Concurrent Quantum Programs

Nengkun Yu and Mingsheng Ying

Tsinghua University, China
University of Technology, Sydney, Australia
{nengkunyu@gmail.com, Mingsheng.Ying@uts.edu.au}

Abstract. We introduce a Markov chain model of concurrent quantum programs. This model is a quantum generalization of Hart, Sharir and Pnueli's probabilistic concurrent programs. Some characterizations of the reachable space, uniformly repeatedly reachable space and termination of a concurrent quantum program are derived by the analysis of their mathematical structures. Based on these characterizations, algorithms for computing the reachable space and uniformly repeatedly reachable space and for deciding the termination are given.

Keywords: Quantum computation, concurrent programs, reachability, termination

1 Introduction

Research on concurrency in quantum computing started about 10 years ago, and it was motivated by two different requirements:

- *Verification of quantum communication protocols:* Quantum communication systems are already commercially available from Id Quantique, MagiQ Technologies, SmartQuantum and NEC. Their advantage over classical communication is that security is provable based on the principles of quantum mechanics. As is well known, it is very difficult to guarantee correctness of even classical communication protocols in the stage of design. Thus, numerous techniques for verifying classical communication protocols have been developed. Human intuition is much better adapted to the classical world than the quantum world. This will make quantum protocol designers to commit many more faults than classical protocol designers. So, it is even more critical to develop formal methods for verification of quantum protocols (see for example [10], [11], [4]). Concurrency is a feature that must be encompassed into the formal models of quantum communication systems.
- *Programming for distributed quantum computing:* A major reason for distributed quantum computing, different from the classical case, comes from the extreme difficulty of the physical implementation of functional quantum computers (see for example [1], [21]). Despite convincing laboratory demonstrations of quantum computing devices, it is beyond the ability of

the current physical technology to scale them. Thus, a natural idea is to use the physical resources of two or more small capacity quantum computers to simulate a large capacity quantum computer. In fact, various experiments in the physical implementation of distributed quantum computing have been frequently reported in recent years. Concurrency naturally arises in the studies of programming for distributed quantum computing.

The majority of work on concurrency in quantum computing is based on process algebras [13], [15], [8], [9], [14], [6], [22], [7], [3]. This paper introduces a new model of concurrent quantum programs in terms of quantum Markov chains. This model is indeed a quantum extension of Hart, Sharir and Pnueli's model of probabilistic concurrent programs [12], [19]. Specifically, a concurrent quantum program consists of a finite set of processes. These processes share a state Hilbert space, and each of them is seen as a quantum Markov chain on the state space. The behaviour of each processes is described by a super-operator. This description of a single process follows Selinger, D'Hont and Panangaden's pioneering works [18], [5] on sequential quantum programs where the denotational semantics of a quantum program is given as a super-operator. The super-operator description of sequential quantum programs was also adopted in one of the authors' work on quantum Floyd-Hoare logic [20]. Similar to the classical and probabilistic cases [12], an execution path of a concurrent quantum program is defined to be an infinite sequence of the labels of their processes, and a certain fairness condition is imposed on an execution path to guarantee that all the processes fairly participate in a computation.

Reachability and termination are two of the central problems in program analysis and verification. The aim of this paper is to develop algorithms that compute the reachable states and decide the termination, respectively, of a concurrent quantum program. To this end, we need to overcome two major difficulties, which are peculiar to the quantum setting and would not arise in the classical case:

- The state Hilbert space of a quantum program is a continuum and thus doomed-to-be infinite even when its dimension is finite. So, a brute-force search is totally ineffective although it may work well to solve a corresponding problem for a classical program. We circumvent the infinity problem of the state space by finding a finite characterization for reachability and termination of a quantum program through a careful analysis of the mathematical structure underlying them.
- The super-operators used to describe the behaviour of the processes are operators on the space of linear operators on the state space, and they are very hard to directly manipulate. In particular, algorithms for computing super-operators are lacking. We adopt a kind of matrix representation for super-operators that allows us to conduct reachability and termination analysis of quantum programs by efficient matrix algorithms.

The paper is organized as follows. For convenience of the reader we briefly recall some basic notions from quantum theory and fix the notations in Sec. 2; but

we refer to [17] for more details. A Markov chain model of concurrent quantum programs is defined in Sec. 3, where we also give a running example of quantum walks. In Sec. 4, we present a characterization for reachable space and one for uniformly repeatedly reachable space of a quantum program, and develop two algorithms to compute them. A characterization of termination of a quantum program with fair execution paths and an algorithm for deciding it are given in Sec. 5. It should be pointed out that termination decision in Sec. 5 is based on reachability analysis in Sec. 4. A brief conclusion is drawn in Sec. 6.

2 Preliminaries and Notations

2.1 Hilbert Spaces

The state space of a quantum system is a Hilbert space. In this paper, we only consider a finite dimensional Hilbert space \mathcal{H} , which is a complex vector space equipped with an inner product $\langle \cdot | \cdot \rangle$. A pure state of a quantum system is represented by a unit vector, i.e., a vector $|\psi\rangle$ with $\langle \psi | \psi \rangle = 1$. Two vectors $|\varphi\rangle, |\psi\rangle$ in \mathcal{H} are orthogonal, written $|\varphi\rangle \perp |\psi\rangle$, if their inner product is 0. A basis of \mathcal{H} is orthonormal if its elements are mutually orthogonal, unit vectors. The trace of a linear operator A on \mathcal{H} is defined to be $\text{tr}(A) = \sum_i \langle i | A | i \rangle$, where $\{|i\rangle\}$ is an orthonormal basis of \mathcal{H} . For a subset V of \mathcal{H} , the subspace $\text{span}V$ spanned by V consists of all linear combinations of vectors in V . For any subspace X of \mathcal{H} , its orthocomplement is the subspace $X^\perp = \{|\varphi\rangle \in \mathcal{H} : |\varphi\rangle \perp |\psi\rangle \text{ for all } |\psi\rangle \in X\}$. The join of a family $\{X_i\}$ of subspaces is $\bigvee_i X_i = \text{span}(\bigcup_i X_i)$. In particular, we write $X \vee Y$ for the join of two subspaces X and Y . A linear operator P is called the projection onto a subspace X if $P|\psi\rangle = |\psi\rangle$ for all $|\psi\rangle \in X$ and $P|\psi\rangle = 0$ for all $|\psi\rangle \in X^\perp$. We write P_X for the projection onto X .

A mixed state of a quantum system is represented by a density operator. A linear operator ρ on \mathcal{H} is called a density operator (resp. partial density operator) if ρ is positive-semidefinite in the sense that $\langle \phi | \rho | \phi \rangle \geq 0$ for all $|\phi\rangle$, and $\text{tr}(\rho) = 1$ (resp. $\text{tr}(\rho) \leq 1$). For any statistical ensemble $\{(p_i, |\psi_i\rangle)\}$ of pure quantum states with $p_i > 0$ for all i and $\sum_i p_i = 1$, $\rho = \sum_i p_i |\psi_i\rangle \langle \psi_i|$ is a density operator. Conversely, each density operator can be generated by an ensemble of pure states in this way. In particular, we write ψ for the density operator $|\psi\rangle \langle \psi|$ generated by a single pure states $|\psi\rangle$. The support of a partial density operator ρ , written $\text{supp}(\rho)$, is the space spanned by its eigenvectors with nonzero eigenvalues.

Lemma 1 *For any $p > 0$ and partial density operators ρ, σ , we have: (1) $\text{supp}(p\rho) = \text{supp}(\rho)$; (2) $\text{supp}(\rho) \subseteq \text{supp}(\rho + \sigma)$; (3) $\text{supp}(\rho + \sigma) = \text{supp}(\rho) \vee \text{supp}(\sigma)$.*

2.2 Super-Operators

A super-operator is a mathematical formalism used to describe a broad class of transformations that a quantum system can undergo. A super-operator on

\mathcal{H} is a linear operator \mathcal{E} from the space of linear operators on \mathcal{H} into itself, satisfying (1) $\text{tr}[\mathcal{E}(\rho)] \leq \text{tr}(\rho)$ for any ρ ; (2) Complete positivity(CP): for any extra Hilbert space \mathcal{H}_k , $(\mathcal{I}_k \otimes \mathcal{E})(A)$ is positive provided A is a positive operator on $\mathcal{H}_k \otimes \mathcal{H}$, where \mathcal{I}_k is the identity operation on \mathcal{H}_k . Furthermore, if $\text{tr}[\mathcal{E}(\rho)] = \text{tr}(\rho)$ for any ρ , then \mathcal{E} is said to be trace-preserving. Each super-operator \mathcal{E} enjoys the Kraus representation: there exists a set of operators $\{E_i\}$ satisfying (1) $\mathcal{E}(\rho) = \sum_i E_i \rho E_i^\dagger$ for all density operators ρ ; (2) $\sum_i E_i^\dagger E_i \leq I$, with equality for trace-preserving \mathcal{E} , where I is the identity operator. In this case, we write $\mathcal{E} = \sum_i E_i \cdot E_i^\dagger$. The image of subspace X of \mathcal{H} under \mathcal{E} is $\mathcal{E}(X) = \bigvee_{|\psi\rangle \in X} \text{supp}(\mathcal{E}(\psi))$, and the pre-image of X under \mathcal{E} is $\mathcal{E}^{-1}(X) = \{|\psi\rangle \in \mathcal{H} : \text{supp}(\mathcal{E}(\psi)) \subseteq X\}$.

Lemma 2 (1) $\text{supp}(\rho) \subseteq \text{supp}(\sigma) \Rightarrow \text{supp}(\mathcal{E}(\rho)) \subseteq \text{supp}(\mathcal{E}(\sigma))$, and $\text{supp}(\rho) = \text{supp}(\sigma) \Rightarrow \text{supp}(\mathcal{E}(\rho)) = \text{supp}(\mathcal{E}(\sigma))$.

(2) $\text{supp}(\mathcal{E}(\rho)) \subseteq \text{supp}((\mathcal{E} + \mathcal{F})(\rho))$. (3) $\mathcal{E}(X) = \text{supp}(\mathcal{E}(P_X))$.

(4) $X \subseteq Y \Rightarrow \mathcal{E}(X) \subseteq \mathcal{E}(Y)$. (5) $\mathcal{E}(X) \subseteq (\mathcal{E} + \mathcal{F})(X)$.

(6) If $\mathcal{E} = \sum_i E_i \cdot E_i^\dagger$, then $\mathcal{E}^{-1}(X) = [\text{supp}(\mathcal{E}^*(P_{X^\perp}))]^\perp$, where $\mathcal{E}^* = \sum_i E_i^\dagger \cdot E_i$ is the (Schrödinger-Heisenberg) dual of \mathcal{E} .

2.3 Matrix Representation of Super-Operator

The matrix representation of a super-operator is usually easier in [24] to manipulate than the super-operator itself. If $\mathcal{E} = \sum_i E_i \cdot E_i^\dagger$ and $\dim \mathcal{H} = d$, then the matrix representation of \mathcal{E} is the $d^2 \times d^2$ matrix $M = \sum_i E_i \otimes E_i^*$, where A^* stands for the conjugate of matrix A , i.e., $A^* = (a_{ij}^*)$ with a_{ij}^* being the conjugate of complex number a_{ij} , whenever $A = (a_{ij})$. According to [24], we have the following

Lemma 3 (1) The modulus of any eigenvalue of M is less or equal to 1.

(2) We write $|\Phi\rangle = \sum_j |jj\rangle$ for the (unnormalized) maximally entangled state in $\mathcal{H} \otimes \mathcal{H}$, where $\{|j\rangle\}$ is an orthonormal basis of \mathcal{H} . Then for any $d \times d$ matrix A , we have $(\mathcal{E}(A) \otimes I)|\Phi\rangle = M(A \otimes I)|\Phi\rangle$.

2.4 Quantum Measurements

A quantum measurement is described by a collection $\{M_m\}$ of operators, where the indexes m refer to the measurement outcomes. It is required that the measurement operators satisfy the completeness equation $\sum_m M_m^\dagger M_m = I_{\mathcal{H}}$. If the system is in state ρ , then the probability that measurement result m occurs is given by $p(m) = \text{tr}(M_m^\dagger M_m \rho)$, and the state of the system after the measurement is $\frac{M_m \rho M_m^\dagger}{p(m)}$.

3 A Model of Concurrent Quantum Programs

Our model is a quantum extension of Hart, Sharir and Pnueli's probabilistic concurrent programs [12]. A concurrent quantum program consists of a finite set

$K = \{1, 2, \dots, m\}$ of quantum processes, and these processes have a common state space, which is assumed to be a d -dimensional Hilbert space \mathcal{H} . With each $k \in K$ we associate a trace-preserving super-operator \mathcal{E}_k , describing a single atomic action or evolution of process k . Also, we assume a termination condition for the program. At the end of each execution step, we check whether this condition is satisfied or not. The termination condition is modeled by a yes-no measurement $\{M_0, M_1\}$: if the measurement outcome is 0, then the program terminates, and we can imagine the program state falls into a terminal (absorbing) space and it remains there forever; otherwise, the program will enter the next step and continues to perform a quantum operation chosen from K .

Definition 1 *A concurrent quantum program defined on a d -dimensional Hilbert space \mathcal{H} is a pair $\mathcal{P} = (\{\mathcal{E}_k : k \in K\}, \{M_0, M_1\})$, where:*

1. \mathcal{E}_k is a super-operator on \mathcal{H} for each $k \in K$;
2. $\{M_0, M_1\}$ is a measurement on \mathcal{H} as the termination test.

Any finite string $s_1 s_2 \dots s_m$ or infinite string $s_1 s_2 \dots s_i \dots$ of elements of K is called a execution path of the program. Thus, the sets of finite and infinite execution paths of program \mathcal{P} are

$$S = K^\omega = \{s_1 s_2 \dots s_i \dots : s_i \in K \text{ for every } i \geq 1\},$$

$$S_{fin} = K^* = \{s_1 s_2 \dots s_m : m \geq 0 \text{ and } s_i \in K \text{ for all } 1 \leq i \leq m\},$$

respectively. A subset of S is usually called a schedule.

For simplicity of presentation, we introduce the notation \mathcal{F}_k for any $k \in K$ which stands for the super-operator defined by $\mathcal{F}_k(\rho) = \mathcal{E}_k(M_1 \rho M_1^\dagger)$ for all density operators ρ . Assume the initial state is ρ_0 . The execution of the program under path $s = s_1 s_2 \dots s_k \dots \in S$ can be described as follows. At the first step, we perform the termination measurement $\{M_0, M_1\}$ on the initial state ρ_0 . The probability that the program terminates; that is, the measurement outcome is 0, is $\text{tr}[M_0 \rho_0 M_0^\dagger]$. On the other hand, the probability that the program does not terminate; that is, the measurement outcome is 1, is $p_1^s = \text{tr}[M_1 \rho_0 M_1^\dagger]$, and the program state after the outcome 1 is obtained is $\rho_1^s = M_1 \rho_0 M_1^\dagger / p_1^s$. We adopt Selinger's normalization convention [18] to encode probability and density operator into a partial density operator $p_1^s \rho_1^s = M_1 \rho_0 M_1^\dagger$. Then this (partial) state is transformed by the quantum operation \mathcal{E}_{s_1} to $\mathcal{E}_{s_1}(M_1 \rho_0 M_1^\dagger) = \mathcal{F}_{s_1}(\rho_0)$. The program continues its computation step by step according to the path s . In general, the $(n+1)$ th step is executed upon the partial density operator $p_n^s \rho_n^s = \mathcal{F}_{s_n} \circ \dots \circ \mathcal{F}_{s_2} \circ \mathcal{F}_{s_1}(\rho_0)$, where p_n^s is the probability that the program does not terminate at the n th step, and ρ_n^s is the program state after the termination measurement is performed and outcome 1 is reported at the n th step. For simplicity, let \mathcal{F}_f denote the super-operator $\mathcal{F}_{s_n} \circ \dots \circ \mathcal{F}_{s_2} \circ \mathcal{F}_{s_1}$ for string $f = s_1 s_2 \dots s_n$. Thus, $p_n^s \rho_n^s = \mathcal{F}_{s[n]}(\rho_0)$, where $s[n]$ is used to denote the head $s_1 s_2 \dots s_n$ for any $s = s_1 s_2 \dots s_n \dots \in S$. The probability that the program terminates in the $(n+1)$ th step is then $\text{tr}(M_0(\mathcal{F}_{s[n]}(\rho_0))M_0^\dagger)$, and the probability that the program does not terminate in the $(n+1)$ th step is $p_{n+1}^s = \text{tr}(M_1(\mathcal{F}_{s[n]}(\rho_0))M_1^\dagger)$.

3.1 Fairness

To guarantee that all the processes in a concurrent program can fairly participate in a computation, a certain fairness condition on its execution paths is needed.

Definition 2 An infinite execution path $s = s_1s_2\dots s_i\dots \in S$ is fair if each process appears infinitely often in s ; that is, for each $k \in K$, there are infinitely many $i \geq 1$ such that $s_i = k$.

We write $F = \{s : s \in S \text{ is fair}\}$ for the schedule of all fair execution paths.

Definition 3 A finite execution path $\sigma = s_1s_2 \cdots s_n \in S_{fin}$ is called a fair piece if each process appears during σ ; that is, for each $k \in K$, there exists $i \leq n$ such that $s_i = k$.

F_{fin} is used to denote the set of all fair pieces: $F_{fin} = \{\sigma : \sigma \in S_{fin} \text{ is a fair piece}\}$. It is obvious that $F = F_{fin}^\omega$; in other words, every fair infinite execution path $s \in F$ can be divided into an infinite sequence of fair pieces: $s = f_1f_2 \cdots f_k \cdots$, where $f_i \in F_{fin}$ for each $i > 0$. The fairness defined above can be generalized by introducing the notion of fairness index, which measures the occurrence frequency of every process in an infinite execution path.

Definition 4 For any infinite execution path $s \in F$, its fairness index $f(s)$ is the minimum, over all processes, of the lower limit of the occurrence frequency of the processes in s ; that is,

$$f(s) = \min_{k \in K} \lim_{t \rightarrow \infty} \inf_{n > t} \frac{s(n, k)}{n},$$

where $s(n, k)$ is the number of occurrences of k in $s[n]$.

For any $\delta \geq 0$, we write F_δ for the set of infinite execution paths whose fairness index is greater than δ : $F_\delta = \{s : s \in S \text{ and } f(s) > \delta\}$. Intuitively, within an infinite execution path in F_δ , each process will be woken up with frequency greater than δ . It is clear that $F_0 \subsetneq F$.

3.2 Running Example

We consider two quantum walks on a circle $C_3 = (V, E)$ with vertices $V = \{0, 1, 2\}$ and edges $E = \{(0, 1), (1, 2), (2, 0)\}$. The first quantum walk $\mathcal{W}_1 = (\{W_1\}, \{M_0, M_1\})$ is given as follows:

- The state space is the 3-dimensional Hilbert space with computational basis $\{|i\rangle | i \in V\}$;
- The initial state is $|0\rangle$; this means that the walk starts at the vertex 0;
- A single step of the walk is defined by the unitary operator:

$$W_1 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 1 & 1 \\ 1 & w & w^2 \\ 1 & w^2 & w \end{pmatrix},$$

where $w = e^{2\pi i/3}$. Intuitively, the probabilities of walking to the left and to the right are both $1/3$, and there is also a probability $1/3$ of not walking.

- The termination measurement $\{M_0, M_1\}$ is defined by

$$M_0 = |2\rangle\langle 2|, \quad M_1 = I_3 - |2\rangle\langle 2|,$$

where I_3 is the 3×3 unit matrix.

The second walk $\mathcal{W}_2 = (\{W_2\}, \{M_0, M_1\})$ is similar to the first one, but its single step is described by unitary operator

$$W_2 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 1 & 1 \\ 1 & w^2 & w \\ 1 & w & w^2 \end{pmatrix}.$$

Then we can put these two quantum walks together to form a concurrent program $\mathcal{P} = (\{W_1, W_2\}, \{M_0, M_1\})$. For example, the execution of this concurrent program according to unfair path $1^\omega \notin F$ is equivalent to a sequential program $(\{W_1\}, \{P_0, P_1\})$; and the execution of \mathcal{P} according to fair path $(12)^\omega \in F$ is as follows: we perform the termination measurement $\{M_0, M_1\}$ on the initial state ρ_0 , then the nonterminating part of the program state is transformed by the super-operator $\mathcal{U}_1 = W_1 \cdot W_1^\dagger$, followed by the termination measurement, and then the application of the super-operator $\mathcal{U}_2 = W_2 \cdot W_2^\dagger$, and this procedure is repeated infinitely many times.

4 Reachability

Reachability is at the centre of program analysis. A state is reachable if some finite execution starting in the initial state ends in it. What concerns us in the quantum case is the subspace of \mathcal{H} spanned by reachable states.

Definition 5 *The reachable space of program $\mathcal{P} = (\{\mathcal{E}_k : k \in K\}, \{M_0, M_1\})$ starting in the initial state ρ_0 is*

$$\mathcal{H}_R = \bigvee_{s \in S, j \geq 0} \text{supp } \mathcal{F}_{s[j]}(\rho_0) = \bigvee_{f \in S_{fin}} \text{supp } \mathcal{F}_f(\rho_0).$$

We have the following closed form characterization of the reachable space.

Theorem 1 $\mathcal{H}_R = \text{supp}(\sum_{i=0}^{d-1} \mathcal{F}^i(\rho_0))$, where $d = \dim \mathcal{H}$ is the dimension of \mathcal{H} , and $\mathcal{F} = \sum_{k \in K} \mathcal{F}_k$.

Proof: We write X for the right-hand side. From Lemma 1, we see that $X = \bigvee \{\text{supp } \mathcal{F}_f(\rho_0) : f \in S_{fin}, |f| < d\}$, where $|f|$ denotes the length of string f . According to the definition of reachable space, we know that $X \subseteq \mathcal{H}_R$. To prove the inverse part $X \supseteq \mathcal{H}_R$, for each $n \geq 0$, we define subspace Y_n as follows: $Y_n := \text{supp}(\sum_{i=0}^n \mathcal{F}^i(\rho_0))$. Due to Lemma 1, we know that $Y_0 \subseteq Y_1 \subseteq \dots \subseteq Y_n \subseteq \dots$. Suppose r is the smallest integer satisfying $Y_r = Y_{r+1}$. We observe that $Y_{n+1} = \text{supp}(\rho + \mathcal{F}(P_{Y_n}))$ for all $n \geq 0$. Then it follows that $Y_n = Y_r$ for all $n \geq r$. On the other hand, we have $Y_0 \subsetneq Y_1 \subsetneq \dots \subsetneq Y_r$.

So, $0 < d_0 < d_1 < \dots < d_r \leq d$, where d_0 is the rank of ρ_0 , and d_i is the dimension of subspace Y_i for $1 \leq i \leq r$. Therefore, we have $r \leq d - 1$ and $Y_{d-1} = Y_r \supseteq Y_n$ for all n . Finally, for any $f \in S_{fin}$, it follows from Lemma 2 that $\text{supp}(\mathcal{F}_f(\rho_0)) \subseteq \text{supp}(\mathcal{F}^{|f|}(\rho_0)) \subseteq Y_{|f|} \subseteq Y_{d-1} = X$. Thus, $\mathcal{H}_R \subseteq X$. ■
Now we are able to present an algorithm computing reachable subspace using matrix representation of super-operators. We define $\mathcal{G} = \sum_{k \in K} \mathcal{F}_k / |K|$.

Algorithm 1: Computing reachable space

input : An input state ρ_0 , and the matrix representation G of \mathcal{G}
output: An orthonormal basis B of \mathcal{H}_R .
 $|x\rangle \leftarrow (I - G/2)^{-1}(\rho_0 \otimes I)|\Phi\rangle$;
 $(* |\Phi\rangle = \sum_j |j_A j_B\rangle)$ is the unnormalized maximally entangled state in $\mathcal{H} \otimes \mathcal{H}$ *)
for $j = 1 : d$ **do**
 $|y_j\rangle \leftarrow \langle j_B | x \rangle$;
end
set of states $B \leftarrow \emptyset$;
integer $l \leftarrow 0$;
for $j = 1 : d$ **do**
 $|z\rangle \leftarrow |y_j\rangle - \sum_{k=1}^l \langle b_k | y_j \rangle |b_k\rangle$;
 if $|z\rangle \neq 0$ **then**
 $l \leftarrow l + 1$;
 $|b_l\rangle \leftarrow |z\rangle / \sqrt{\langle z | z \rangle}$;
 $B \leftarrow B \cup \{|b_l\rangle\}$;
 end
end
return

Theorem 2 *Algorithm 1 computes the reachable space in time $\mathcal{O}(d^{4.7454})$, where $d = \dim \mathcal{H}$.*

Proof: It follows from Lemma 3(1) that $I - G/2$ is invertible, and $\sum_{i=0}^{\infty} (G/2)^i = (I - G/2)^{-1}$. We write $\rho = \sum_{i=0}^{\infty} \mathcal{G}^i(\rho_0)/2^i$, and have

$$(\rho \otimes I)|\Phi\rangle = \sum_{i=0}^{\infty} (G/2)^i (\rho_0 \otimes I)|\Phi\rangle,$$

and the existence of ρ immediately follows from Lemma 3. We further see that $|x\rangle = (\rho \otimes I)|\Phi\rangle = \sum_j \rho|j_A\rangle|j_B\rangle$ and $|y_i\rangle = \rho|j_A\rangle$. Note that B is obtained from $\{|y_j\rangle\}$ by the Gram-Schmidt procedure. So, $\text{supp}(\rho) = \text{span}\{\rho|j\rangle\} = \text{span}B$. It is clear that $\mathcal{H}_R = \text{supp}(\sum_{i=0}^{d-1} \mathcal{F}^i(\rho_0)) \subseteq \text{supp}(\rho)$. Therefore, $\mathcal{H}_R = \text{supp}(\rho) = \text{span}B$, and the algorithm is correct.

The complexity comes from three the following parts: (1) it costs $\mathcal{O}(d^{2*2.3727})$ to compute $(I - G/2)^{-1}$ by using Coppersmith-Winograd algorithm [2]; (2) it

requires $\mathcal{O}(d^4)$ to obtain $|x\rangle$ from $(I - G/2)^{-1}$; (3) the Gram-Schmidt orthonormalization is in time $\mathcal{O}(d^3)$. So, the time complexity is $\mathcal{O}(d^{4.7454})$ in total. ■

An advantage of Algorithm 1 is that we can store $(I - G/2)^{-1}$. Then for any input state ρ_0 , we only need $\mathcal{O}(d^4)$ to compute the space reachable from ρ_0 .

Definition 6 *The uniformly repeatedly reachable space of program $\mathcal{P} = (\{\mathcal{E}_k : k \in K\}, \{M_0, M_1\})$ starting in the initial state ρ_0 is*

$$\mathcal{H}_{URR} = \bigcap_{n \geq 0} \bigvee_{s \in S, j \geq n} \text{supp } \mathcal{F}_{s[j]}(\rho_0) = \bigcap_{n \geq 0} \bigvee \{\text{supp } \mathcal{F}_f(\rho_0) : f \in S_{fin}, |f| \geq n\}.$$

The uniformly repeatedly reachable space enjoys the following closed form,

Theorem 3 $\mathcal{H}_{URR} = \text{supp}(\sum_{i=d}^{2d-1} \mathcal{F}^i(\rho_0))$, where $d = \dim \mathcal{H}$, and $\mathcal{F} = \sum_{k \in K} \mathcal{F}_k$.

Proof: For each $n \geq 0$, we define subspace Z_n as follows: $Z_n := \bigvee_{j \geq n} \text{supp } \mathcal{F}^j(\rho_0)$. It is obvious that $Z_0 \supseteq Z_1 \supseteq \dots \supseteq Z_n \supseteq \dots$. Suppose r is the smallest integer satisfying $Z_r = Z_{r+1}$. By noting that $Z_{n+1} = \text{supp}(\mathcal{F}(P_{Z_n}))$, we can show that $Z_n = Z_r$ for all $n \geq r$. On the other hand, we have $Z_0 \supsetneq Z_1 \supsetneq \dots \supsetneq Z_r$. So, $d_0 > d_1 > \dots > d_r \geq 0$, and d_i is the dimension of subspace Z_i for $0 \leq i \leq r$. Therefore, we have $r \leq d_0 \leq d$ and $Z_d = Z_r$. Therefore, $\mathcal{H}_{URR} = \bigcap_{n \geq 0} Z_n = Z_d$. It is obvious that Z_d is the reachable space starting in state $\mathcal{F}^d(\rho_0)$. Using Theorem 1 we obtain $Z_d = \text{supp}(\sum_{i=0}^{d-1} \mathcal{F}^i(\mathcal{F}^d(\rho_0))) = \text{supp}(\sum_{i=d}^{2d-1} \mathcal{F}^i(\rho_0))$. ■

We can give an algorithm computing the uniformly repeatedly reachable space by combining the above theorem and matrix representation of super-operators.

Algorithm 2: Compute uniformly repeatedly reachable space

input : An input state ρ_0 , and the matrix representation G of \mathcal{G}
output: An orthonormal basis B_{URR} of \mathcal{H}_{URR} .
 $|x\rangle \leftarrow G^d(I - G/2)^{-1}(\rho_0 \otimes I)|\Phi\rangle$;
 (* $|\Phi\rangle = \sum_j |j_A j_B\rangle$ is the unnormalized maximally entangled state in $\mathcal{H} \otimes \mathcal{H}$ *)
for $j = 1 : d$ **do**
 | $|y_j\rangle \leftarrow \langle j_B | x \rangle$;
end
set of states $B_{URR} \leftarrow \emptyset$;
integer $l \leftarrow 0$;
for $j = 1 : d$ **do**
 | $|z\rangle \leftarrow |y_j\rangle - \sum_{k=1}^l \langle b_k | y_j \rangle |b_k\rangle$;
 | **if** $|z\rangle \neq 0$ **then**
 | | $l \leftarrow l + 1$;
 | | $|b_l\rangle \leftarrow |z\rangle / \sqrt{\langle z | z \rangle}$;
 | | $B_{URR} \leftarrow B_{URR} \cup \{|b_l\rangle\}$;
 | **end**
end
return

Theorem 4 *Algorithm 2 computes the uniformly repeatedly reachable space in time $\mathcal{O}(d^{4.7454} \log d)$, where $d = \dim \mathcal{H}$.*

Proof: This theorem is a corollary of Theorem 2. Here, $\log d$ in the complexity comes from computing M^d using the method of exponentiation by squaring.

5 Termination

Another important problem concerning the behaviour of a program is its termination.

Definition 7 *Let the program $\mathcal{P} = (\{\mathcal{E}_k : k \in K\}, \{M_0, M_1\})$. Then \mathcal{P} with input ρ_0 terminates for execution path $s \in S$ if $p_n^s \rho_n^s = \mathcal{F}_{s[n]} = 0$ for some positive integer n .*

Definition 8 1. *If a program \mathcal{P} with input ρ_0 terminates for all $s \in A$, then we say that it terminates in schedule A .*
 2. *If there is a positive integer n such that $p_n^s \rho_n^s = 0$ for all $s \in A$, then it is said that the program \mathcal{P} with input ρ_0 uniformly terminates in schedule A .*

We first prove the equivalence between termination and uniform termination. Of course, this equivalence comes from finiteness of the dimension of the state space.

Theorem 5 *The program $\mathcal{P} = (\{\mathcal{E}_k : k \in K\}, \{M_0, M_1\})$ with initial state ρ_0 terminates in the biggest schedule $S = K^\omega$ if and only if it uniformly terminates in schedule S .*

Proof. The “if” part is obvious. We prove the “only if” part in two steps:

(1) We consider the case of $|K| = 1$, where $\{\mathcal{E}_k : k \in K\}$ is a singleton $\{\mathcal{E}\}$. Now the program is indeed a sequential program, and it is a quantum loop [23]. We write $\mathcal{F}(\rho) = \mathcal{E}(M_1 \rho M_1^\dagger)$ for all ρ . What we need to prove is that if \mathcal{P} terminates, i.e., $\mathcal{F}^n(\rho_0) = 0$ for some n , then it terminates within d steps, i.e., $\mathcal{F}^d(\rho_0) = 0$. If ρ_0 is a pure state $|\psi\rangle$, then we define the termination sets as follows: $X_n := \{|\psi\rangle : \mathcal{F}^n(\psi) = 0\}$ for each integer $n > 0$.

(1.1) If $|\varphi\rangle, |\chi\rangle \in X_n$, then $\mathcal{F}^n(\varphi + \chi) = 0$, which leads to $\alpha|\varphi\rangle + \beta|\chi\rangle \in X_n$ for any $\alpha, \beta \in \mathbb{C}$. Thus X_n is a subspace of \mathcal{H} .

(1.2) Since $\mathcal{F}^n(\psi) = 0 \Rightarrow \mathcal{F}^{n+1}(\psi) = 0$, it holds that $X_n \subseteq X_{n+1}$ for any $n > 0$. So, we have the inclusion relation $X_1 \subseteq X_2 \subseteq \dots \subseteq X_n \subseteq \dots$.

Now suppose t is the smallest integer satisfying $X_t = X_{t+1}$. Invoking Lemma 2, we obtain that $\text{supp}(\mathcal{F}^{*t}(I)) = X_t^\perp = X_{t+1}^\perp = \text{supp}(\mathcal{F}^{*(t+1)}(I))$, where $\mathcal{F}^*(\cdot)$ denotes the (Schrödinger-Heisenberg) dual of $\mathcal{F}(\cdot)$. We have $\text{supp}(\mathcal{F}^{*n}(I)) = \text{supp}(\mathcal{F}^{*t}(I))$, which leads to $X_n = X_t$ for all $n \geq t$. Now, it holds that $X_1 \subsetneq X_2 \subsetneq \dots \subsetneq X_t = X_{t+1} = X_{t+2} = \dots$. This implies $d_1 < d_2 < \dots < d_t$, where d_i is the dimension of subspace X_i . Thus, $t \leq d$. If $\mathcal{F}^n(\psi) = 0$, then $|\psi\rangle \in X_n \subseteq X_d$, and $\mathcal{F}^d(\psi) = 0$.

In general, if ρ_0 is a mixed input state $\rho_0 = \sum p_i |\psi_i\rangle\langle\psi_i|$ with all $p_i > 0$, and $\mathcal{F}^n(\rho_0) = 0$, then $\mathcal{F}^n(\psi_i) = 0$ for all i . Therefore, $\mathcal{F}^d(\psi_i) = 0$ for all i , and it follows immediately that $\mathcal{F}^d(\rho_0) = 0$.

(2) For the general case of $|K| \geq 2$, we assume that \mathcal{P} starting in ρ_0 terminates in S , i.e., for any $s \in S$, there exists an integer n_s such that $\mathcal{F}_{s[n_s]}(\rho_0) = 0$. Our purpose is to show that there exists an integer n such that $\mathcal{F}_{s[n]}(\rho_0) = 0$ for all $s \in S$. Indeed, we can choose $n = d$. We do this by refutation. Assume that $\mathcal{F}_{s[d]}(\rho_0) \neq 0$ for some $s \in S$. We are going to construct an execution path $s \in S$ such that $\mathcal{F}_{s[n]}(\rho_0) \neq 0$ for any $n \geq 0$. Let $\mathcal{F} = \sum_{k \in K} \mathcal{F}_k$. Then the assumption means that there exist $f \in K^d$ such that $\mathcal{F}_f(\rho_0) \neq 0$, and it follows that $\mathcal{F}^d(\rho_0) \neq 0$. Now we consider the loop program $(\{\mathcal{F}\}, \{M_0, M_1\})$ with initial state ρ_0 . Applying (1) to it, we obtain $\mathcal{F}^{2d}(\rho_0) \neq 0$. Then there exist $g_1, h_1 \in K^d$ such that $\mathcal{F}_{h_1}(\mathcal{F}_{g_1}(\rho_0)) = \mathcal{F}_{g_1 h_1}(\rho_0) \neq 0$, and $\mathcal{F}^d(\mathcal{F}_{g_1}(\rho_0)) \neq 0$. Applying (1) again leads to $\mathcal{F}^{2d}(\mathcal{F}_{g_1}(\rho_0)) \neq 0$, which means that there exist $h_2, g_2 \in K^{2d}$ such that $\mathcal{F}_{h_2}(\mathcal{F}_{g_1 g_2}(\rho_0)) = \mathcal{F}_{g_2 h_2}(\mathcal{F}_{g_1}(\rho_0)) \neq 0$. Thus, we have $\mathcal{F}^d(\mathcal{F}_{g_2 g_1}(\rho_0)) \neq 0$. Repeating this procedure, we can find an infinite sequence $g_1, g_2, \dots \in K^d$. Put $s = g_1 g_2 \dots \in S$. Then it holds that $\mathcal{T}_{s[kd]}(\rho_0) \neq 0$ for any integer k . Thus, we have $\mathcal{T}_{s[n]}(\rho) \neq 0$ for all n . ■

Now we are ready to consider termination under fairness. Of course, any permutation of K is a fair piece. We write P_K for the set of permutations of K . For $\sigma = s_1 s_2 \dots s_m \in P_K$, a finite execution path of the form $s_1 \sigma_1 s_2 \sigma_2 \dots \sigma_{m-1} s_m$ is called an expansion of σ . Obviously, for any $\sigma \in P_K$, all of its expansions are in F_{fin} . We will use a special class of fair pieces generated by permutations:

$$\Pi = \{s_1 \sigma_1 s_2 \sigma_2 \dots \sigma_{m-1} s_m : s_1 s_2 \dots s_m \in P_K \text{ and } |\sigma_i| < d \text{ for every } 1 \leq i < m\},$$

where d is the dimension of the Hilbert space \mathcal{H} of program states. It is easy to see that $\Pi \subsetneq F_{fin}$.

Theorem 6 *A program $\mathcal{P} = (\{\mathcal{E}_k : k \in K\}, \{M_0, M_1\})$ with initial state ρ_0 terminates in the fair schedule F if and only if it terminates in the schedule Π^ω .*

Proof. The “only if” part is clear because $\Pi^\omega \subseteq F$. To prove the “if” part, assume \mathcal{P} terminates in the schedule Π^ω . We proceed in four steps:

(1) Since Π is a finite set, we can construct a new program $\mathcal{P}' = (\{\mathcal{F}_f : f \in \Pi\}, \{0, I\})$. (We should point out that \mathcal{F}_f is usually not trace-preserving, and thus \mathcal{P}' is indeed not a program in the sense of Definition 1. However, this does not matter for the following arguments.) It is easy to see that the termination of \mathcal{P} with ρ_0 in schedule Π^ω implies the termination of \mathcal{P}' with ρ_0 in Π^ω . Note that Π^ω is the biggest schedule in \mathcal{P}' , although it is not the biggest schedule in \mathcal{P} . So, we can apply Theorem 5 to \mathcal{P}' and assert that $(\sum_{f \in \Pi} \mathcal{F}_f)^d(\rho_0) = 0$. That is equivalent to

$$\text{supp}[(\sum_{f \in \Pi} \mathcal{F}_f)^d(\rho_0)] = \{0\} \text{ (0-dimensional subspace)}. \quad (1)$$

(2) For each $\sigma \in P_K$, we set $A_\sigma = \{\sigma' \in \Pi : \sigma' \text{ is an expansion of } \sigma\}$. Then $\bigcup_{\sigma \in P_K} A_\sigma = \Pi$. Moreover, we write $\mathcal{G}_\sigma = \sum_{f \in A_\sigma} \mathcal{F}_f$ for every $\sigma \in P_K$. It is worth noting that $\sum_{\sigma \in P_K} \mathcal{G}_\sigma = \sum_{f \in \Pi} \mathcal{F}_f$ is not true in general because it is possible that $A_{\sigma_1} \cap A_{\sigma_2} \neq \emptyset$ for different σ_1 and σ_2 . But by Lemma 1.1 and 3 we have

$$\text{supp}[(\sum_{\sigma \in P_K} \mathcal{G}_\sigma)(\rho_0)] = \text{supp}[(\sum_{f \in \Pi} \mathcal{F}_f)(\rho_0)],$$

and furthermore, it follows from Eq. (1) that

$$\text{supp}[(\sum_{\sigma \in P_K} \mathcal{G}_\sigma)^d(\rho_0)] = \text{supp}[(\sum_{f \in \Pi} \mathcal{F}_f)^d(\rho_0)] = \{0\}. \quad (2)$$

(3) For each fair piece $\sigma' \in F_{fin}$, and for any ρ , we can write $\sigma' = s_1 f_1 s_2 \cdots s_{m-1} f_{m-1} s_m$ for some $\sigma_0 = s_1 s_2 \cdots s_m \in P_K$, and $f_1, \dots, f_{m-1} \in S_{fin}$. Furthermore, we write $\mathcal{G} = \sum_{i=0}^{d-1} (\sum_{k=1}^m \mathcal{F}_k)^i$. First, a routine calculation leads to $\mathcal{G}_{\sigma_0} = \mathcal{F}_{s_m} \circ \mathcal{G} \circ \mathcal{F}_{s_{m-1}} \cdots \mathcal{F}_{s_2} \circ \mathcal{G} \circ \mathcal{F}_{s_1}$. Second, it follows from Theorem 1 that for each $1 \leq i \leq m-1$, and for any ρ , $\text{supp}(\mathcal{F}_{f_i}(\rho)) \subseteq \text{supp}(\mathcal{G}(\rho))$. Repeatedly applying this inclusion together with Lemma 2.1 we obtain

$$\begin{aligned} \text{supp}(\mathcal{F}_{\sigma'}(\rho)) &= \text{supp}[(\mathcal{F}_{s_m} \circ \mathcal{F}_{f_{m-1}} \circ \mathcal{F}_{s_{m-1}} \circ \cdots \circ \mathcal{F}_{s_2} \circ \mathcal{F}_{f_1} \circ \mathcal{F}_{s_1})(\rho)] \\ &\subseteq \text{supp}[(\mathcal{F}_{s_m} \circ \mathcal{G} \circ \mathcal{F}_{s_{m-1}} \circ \cdots \circ \mathcal{F}_{s_2} \circ \mathcal{G} \circ \mathcal{F}_{s_1})(\rho)] \\ &= \text{supp}(\mathcal{G}_{\sigma_0}(\rho)) \subseteq \text{supp}(\sum_{\sigma \in \Pi} \mathcal{F}_\sigma(\rho)). \end{aligned} \quad (3)$$

(4) Now we are able to complete the proof by showing that for any fair execution path $s \in F$, s has an initial segment t such that $\mathcal{F}_t(\rho_0) = 0$. In fact, s can be written as an infinite sequence of fair piece, i.e., $s = \sigma'_1 \sigma'_2 \cdots$, where each σ'_i is a fair piece. We take t to be the initial segment of s containing the first d fair pieces, i.e., $t = \sigma_1 \sigma_2 \cdots \sigma_d$. Repeatedly applying Eq. (3) and Lemma 2.1 we obtain

$$\begin{aligned} \text{supp}\mathcal{F}_t(\rho_0) &= \text{supp}[(\mathcal{F}_{\sigma'_d} \circ \cdots \circ \mathcal{F}_{\sigma'_2} \circ \mathcal{F}_{\sigma'_1})(\rho_0)] \\ &\subseteq \text{supp}[(\sum_{p \in \Pi} \mathcal{F}_p)^d(\rho)] = \{0\}. \end{aligned}$$

Thus, $\mathcal{F}_t(\rho) = 0$. ■

The above theorem can be slightly strengthened by employing the notion of fairness index in Definition 4. First, we have:

Lemma 4 $\Pi^\omega \subsetneq F_{\frac{1}{md}}$.

Proof. For any $s = \sigma_1 \sigma_2 \cdots \in \Pi^\omega$ with $\sigma_i \in \Pi$, we know that $\sigma_i(|\sigma_i|, k) \geq 1$ for any $k \in K$ and $|\sigma_i| < md$, where $\sigma_i(|\sigma_i|, k)$ is the number of occurrences of k in σ_i . Then the occurrence frequency $f(s) > \frac{1}{md}$, which means that $\Pi^\omega \subseteq F_{\frac{1}{md}}$. On the other hand, we choose an arbitrary $s \in F_{\frac{1}{md}}$. Then $1^{md}s \in F_{\frac{1}{md}}$ but $1^{md}s \notin \Pi^\omega$. ■

Actually, what we proved in Theorem 6 is that for any two schedules A, B between Π^ω and F , i.e., $\Pi^\omega \subset A, B \subset F$, a program terminates in schedule A if and only if it terminates in schedule B . Combining Theorem 6 and Lemma 4 yields:

Corollary 1 *For any $0 \leq \delta, \epsilon \leq \frac{1}{md}$, a program terminates in schedule F_δ if and only if it terminates in schedule F_ϵ . ■*

Now an algorithm checking termination of a concurrent quantum program can be developed based on Theorem 5.

Algorithm 3: Decide termination of a concurrent quantum program

```

input : An input state  $\rho_0$ , and the matrix representation of each  $\mathcal{F}_i$  i.e,  $N_i$ 
output: b.(If the program terminates under  $F$ ,  $b = 0$ ; Otherwise,  $b = 1$ .)
 $N \leftarrow 0$ ;
for  $k = 1 : m$  do
     $N \leftarrow N_i + N$ ;
end
 $G \leftarrow I$ ;
for  $k = 1 : d - 1$  do
     $G \leftarrow I + NG$ ;
end
(*Compute the matrix representation of  $\mathcal{G}^*$ )
 $M \leftarrow 0$ ;
Generate  $P_K$ ;
for  $p = p_1 p_2 \cdots p_m \in P_K$  do
     $L \leftarrow N_{p_1}$ ;
    for  $l = 2 : m$  do
         $L \leftarrow N_{p_l} G L$ ;
    end
    (*Compute the matrix representation of  $\mathcal{F}_p^*$ )
     $M \leftarrow M + L$ ;
end
(*Compute the matrix representation of  $\sum_{p \in P_K} \mathcal{F}_p^*$ )
 $|x\rangle \leftarrow M^d(\rho_0 \otimes I)|\Phi\rangle$ ;
if  $|x\rangle \neq 0$  then
     $b \leftarrow 1$ ;
end
if  $|x\rangle = 0$  then
     $b \leftarrow 0$ ;
end
return b

```

Theorem 7 *Algorithm 3 decides termination of a concurrent quantum program in time $\mathcal{O}(m^m d^{4.7454})$, where m is the number of the processes, and $d = \dim \mathcal{H}$.*

Proof: In the algorithm, we use the for loop to compute the matrix representation G of $\mathcal{G} = \sum_{i=0}^{d-1} (\sum_{k=1}^m \mathcal{F}_k)^i$. Then the matrix representation of $\mathcal{F}_\sigma = \mathcal{F}_{s_1} \circ \mathcal{G} \circ \mathcal{F}_{s_2} \cdots \mathcal{G} \circ \mathcal{F}_{s_m}(\cdot)$ is obtained for any $\sigma = s_1 s_2 \cdots s_m \in P_K$. All \mathcal{F}_σ s are added up to M . Then M becomes the matrix representation of $\sum_{\sigma \in P_K} \mathcal{F}_\sigma$. Consequently, we can apply Theorem 6 to assert that this algorithm outputs 0 if the program terminates in the fair schedule F ; otherwise, 1.

To analyse its complexity, the algorithm can be divided into three steps: (1) Computing G costs $\mathcal{O}(m+d \cdot d^{2*2.3727}) = \mathcal{O}(m+d^{5.7454})$; (2) Computing M costs $m! * 2m * \mathcal{O}(d^{2*2.3727}) = \mathcal{O}(m^m \cdot d^{4.7454})$; (3) Computing $|x\rangle$ costs $\mathcal{O}(d^{4.7454} \log d)$. So the total cost is $\mathcal{O}((m^m + d)d^{4.7454})$. ■

6 Conclusion

In this paper, we studied two of the central problems, namely, reachability and termination for concurrent quantum programs. A concurrent quantum program is modeled by a family of quantum Markov chains sharing a state Hilbert space and a termination measurement, with each chain standing for a participating process. This model extends Hart, Sharir and Pnueli's model of probabilistic concurrent programs [12] to the quantum setting. We show that the reachable space and the uniformly repeatedly reachable space of a concurrent quantum program can be computed and its termination can be decided in time $\mathcal{O}(d^{4.7454})$, $\mathcal{O}(d^{4.7454} \log d)$, $\mathcal{O}((m^m + d)d^{4.7454})$, respectively, where m is the number of participating processes, and d is the dimension of state space.

For further studies, an obvious problem is: how to improve the above algorithm complexities? In this paper, reachability and termination of quantum programs were defined in a way where probabilities are abstracted out; that is, only reachability and termination with certainty are considered. A more delicate, probability analysis of the reachability and termination is also an interesting open problem. The algorithms for computing the reachable space and checking termination of a *quantum* program presented in this paper are all algorithms for *classical* computers. So, another interesting problem is to find efficient *quantum* algorithms for reachability and termination analysis of a quantum program.

Acknowledgment

We are grateful to Dr Yangjia Li, Runyao Duan and Yuan Feng for useful discussions. This work was partly supported by the Australian Research Council (Grant No. DP110103473).

References

1. J. I. Cirac, A. K. Ekert, S. F. Huelga and C. Macchiavello, Distributed quantum computation over noisy channels, *Physical Review A* 59(1999)4249-4254.
2. C. Don and W. Shmuel, Matrix multiplication via arithmetic progressions, *Journal of Symbolic Computation* 9(1990)251-280.

3. T. A. S. Davidson, *Formal Verification Techniques using Quantum Process Calculus*, Ph.D. thesis, University of Warwick, 2011.
4. T. Davidson, S. Gay, R. Nagarajan and I. V. Puthoor, Analysis of a quantum error correcting code using quantum process calculus, *Proceedings of QPL 2011, the 8th Workshop on Quantum Physics and Logic*, pp. 107-120.
5. E. D'Hondt and P. Panangaden, Quantum weakest preconditions, *Mathematical Structures in Computer Science* 16(2006)429-451.
6. Y. Feng, R. Y. Duan, Z. F. Ji and M. S. Ying, Probabilistic bisimulations for quantum processes, *Information and Computation* 205(2007)1608-1639.
7. Y. Feng, R. Y. Duan and M. S. Ying, Bisimulation for quantum processes, *Proceedings of the 38th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, 2011, pp. 523-534.
8. S. J. Gay and R. Nagarajan, Communicating Quantum Processes, *Proceedings of the 32nd ACM Symposium on Principles of Programming Languages (POPL)*, 2005, pp. 145-157.
9. S. J. Gay and R. Nagarajan, Types and typechecking for communicating quantum processes, *Mathematical Structures in Computer Science* 16(2006)375D406.
10. S. J. Gay, N. Papanikolaou and R. Nagarajan, QMC: a model checker for quantum systems. *Proceedings of the 20th International Conference on Computer Aided Verification (CAV)*, 2008, Springer LNCS 5123, pp. 543-547.
11. S. J. Gay, N. Papanikolaou and R. Nagarajan, Specification and verification of quantum protocols, *Semantic Techniques in Quantum Computation* (S. J. Gay and I. Mackie, eds.), Cambridge University Press, 2010, pp. 414-472.
12. S. Hart, M. Sharir and A. Pnueli, Termination of probabilistic concurrent programs, *ACM Transactions on Programming Languages and Systems* 5(1983)356-380.
13. P. Jorrand and M. Lalire, Toward a quantum process algebra, *Proceedings of the First ACM Conference on Computing Frontiers*, 2004, pp. 111-119.
14. M. Lalire, Relations among quantum processes: bisimilarity and congruence, *Mathematical Structures in Computer Science* 16(2006)407-428.
15. M. Lalire and P. Jorrand, A process algebraic approach to concurrent and distributed quantum computation: operational semantics, *Proceedings of the 2nd International Workshop on Quantum Programming Languages*, 2004.
16. Y. Y. Li, N. K. Yu and M. S. Ying, Termination of nondeterministic quantum programs, Short presentation of *LICS'2012* (For full paper, see *arXiv*: 1201.0891).
17. M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, Cambridge, 2000.
18. P. Selinger, Towards a quantum programming language, *Mathematical Structure in Computer Science* 14(2004)527-586.
19. M. Sharir, A. Pnueli and S. Hart, Verification of probabilistic programs, *SIAM Journal on Computing* 13(1984)292-314.
20. M. S. Ying, Floyd-Hoare logic for quantum programs, *ACM Transactions on Programming Languages and Systems* 33(2011) art. no. 19.
21. M. S. Ying and Y. Feng, An algebraic language for distributed quantum computing, *IEEE Transactions on Computers* 58(2009)728-743.
22. M. S. Ying, Y. Feng, R. Y. Duan and Z. F. Ji, An algebra of quantum processes, *ACM Transactions on Computational Logic* 10(2009) art. no. 19.
23. M. S. Ying and Y. Feng, Quantum loop programs, *Acta Informatica* 47(2010)221-250.
24. M. S. Ying, N. K. Yu, Y. Feng and R. Y. Duan, Verification of Quantum Programs, *arXiv*:1106.4063.